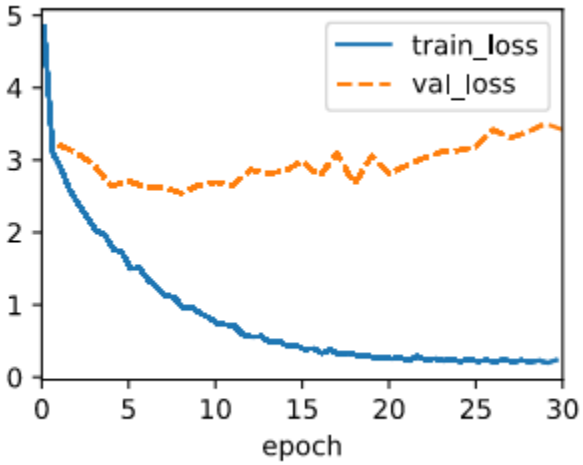
ECGR-5103: Homework 5
Patrick Flynn | ID: 801055057
18 March 2023
Github: https://github.com/pflynn157/ecgr-5106

Problem 1

Overall, the results from the sequence-to-sequence model within this part were fairly consistent. As we shall see, adjusting some aspects of the model aided in improving the accuracy and reducing loss, but unlike previous problems (in previous homeworks), the differences were not as drastic. In this first part, we start with the baseline model, adjust the number of layers, and then replace the GRU with LSTM.

The baseline sequence-to-sequence model yielded a low training loss as the number of epochs increased, but the validation loss increased significantly.

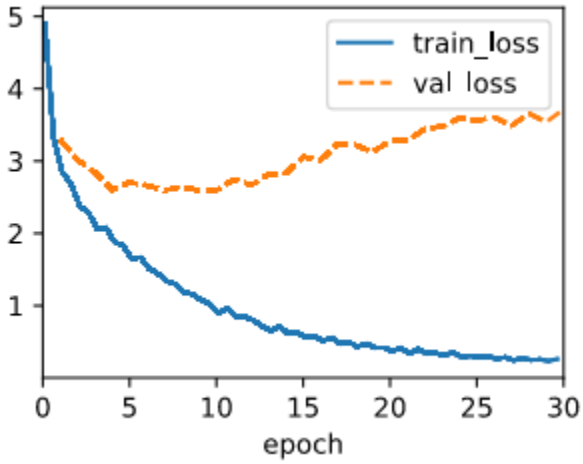


The initial prediction results:

```
go . => ['va', '!'], bleu,1.000
i lost . => ["j'ai", 'perdu', '.'], bleu,1.000
he's calm . => ['<unk>', 'confiance', '.'], bleu,0.000
i'm home . => ['je', 'suis', 'chez', 'moi', '.'], bleu,1.000
```

In our first adjustment, we adjusted the number of layers to 3 on the encoder and 2 on the decoder, thereby creating a staggered number of layers between the two components. This did not make a huge difference. The validation and training losses were roughly the same, though the validation loss increased more gradually than did originally.

Here are the results:

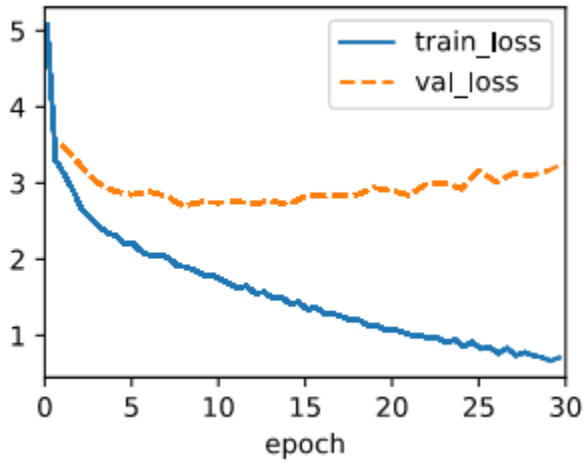


And the prediction results, which were the same:

```
go . => ['va', '!'], bleu,1.000
i lost . => ["j'ai", 'perdu', '.'], bleu,1.000
he's calm . => ['<unk>', 'confiance', '.'], bleu,0.000
i'm home . => ['je', 'suis', 'chez', 'moi', '.'], bleu,1.000
```

In the next part, we replaced the GRU model with LSTM. The results here were interesting. The training loss ultimately went as low as the previous experiments, but it took longer for the loss to drop. The validation loss was also lower than the two previous experiments. However, the prediction results were terrible.

The training results:



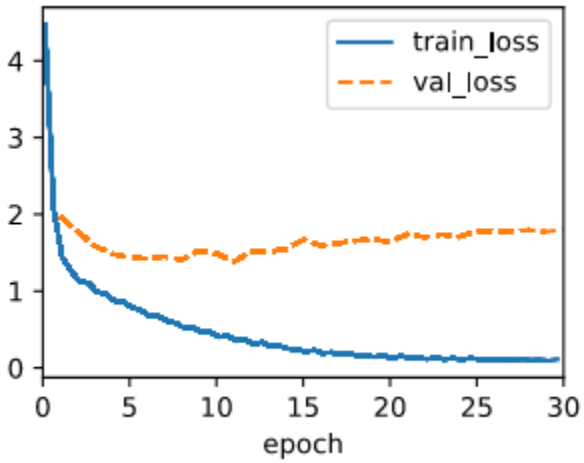
The prediction results:

```
go . => ['vas-y', '.'], bleu,0.000
i lost . => ['je', 'suis', 'parti', '.'], bleu,0.000
he's calm . => ["c'est", '<unk>', '.'], bleu,0.000
i'm home . => ['je', 'suis', 'calme', '.'], bleu,0.512
```

Problem 2

In this section, we used the Bahdanu attention based sequence-to-sequence modeling. Overall, this model seemed to do better than the regular sequence-to-sequence model. In this section, we did experiments on the number of layers, and then replaced GRU with LSTM.

Here is the baseline model:

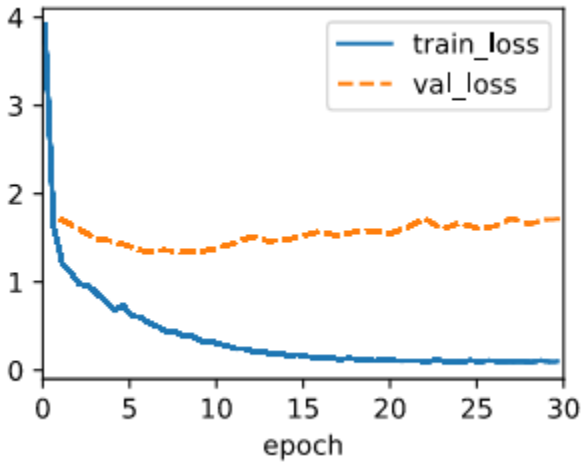


And the prediction results:

```
go . => ['va', '!'], bleu,1.000
i lost . => ["j'ai", 'perdu', '.'], bleu,1.000
he's calm . => ['il', 'est', 'mouillé', '.'], bleu,0.658
i'm home . => ['je', 'suis', 'chez', 'moi', '.'], bleu,1.000
```

In the following experiments, we adjusted the number of layers from 1-4. The results here were varied. Overall, the training and validation losses did not change that much, though the validation losses did fluctuate some across the experiments. The prediction results did change significantly though. The test with 3 layers yielded the best predictions, whereas the test with 4 layers yielded the worst.

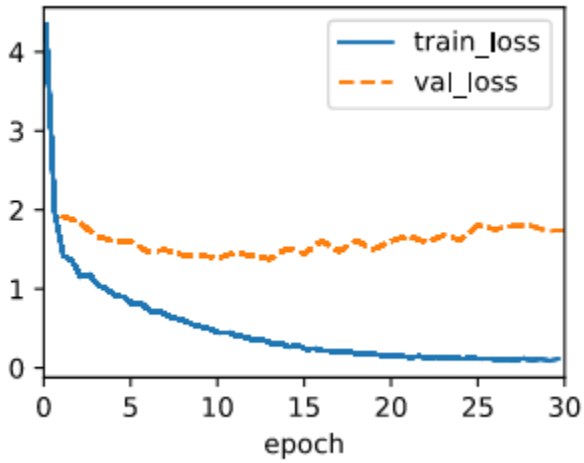
With 1 layer.
The training results:



The prediction results:

```
go . => ['va', '!'], bleu,1.000
i lost . => ["j'ai", 'perdu', '.'], bleu,1.000
he's calm . => ['<unk>', '.'], bleu,0.000
i'm home . => ['je', 'suis', 'chez', 'moi', '.'], bleu,1.000
```

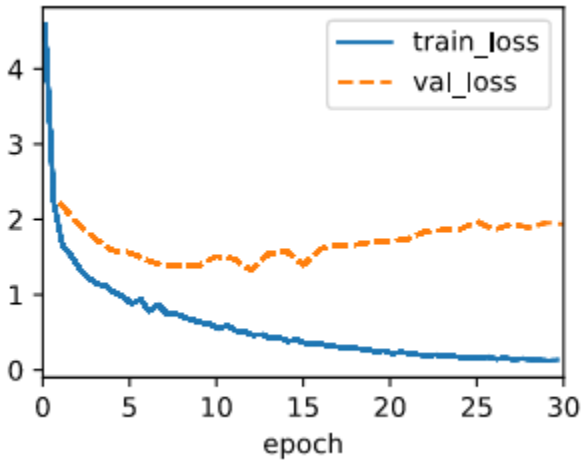
With 2 layers:
The training results:



The prediction results:

```
go . => ['va', '!'], bleu,1.000
i lost . => ["j'ai", 'perdu', '.'], bleu,1.000
he's calm . => ['je', 'vais', 'bien', '.'], bleu,0.000
i'm home . => ['je', 'suis', 'chez', 'moi', '.'], bleu,1.000
```

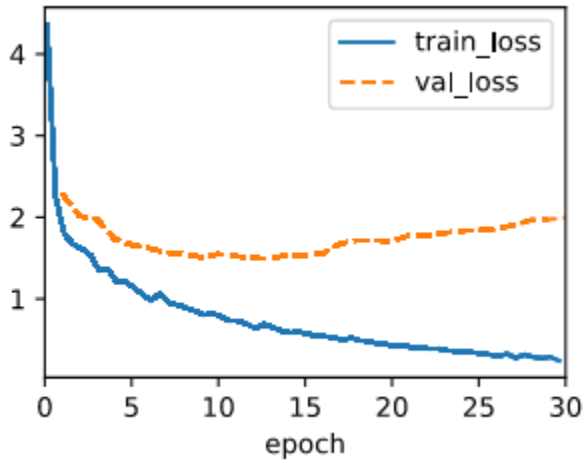
With 3 layers:
The training results:



The prediction results:

```
go . => ['va', '!'], bleu,1.000
i lost . => ["j'ai", 'perdu', '.'], bleu,1.000
he's calm . => ['il', 'est', 'mouillé', '.'], bleu,0.658
i'm home . => ['je', 'suis', 'chez', 'moi', '.'], bleu,1.000
```

With 4 layers:
The training results:

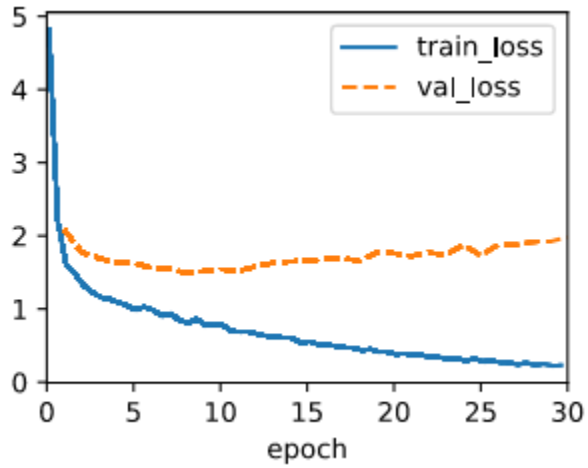


The prediction results:

```
go . => ['<unk>', '!'], bleu,0.000
i lost . => ["j'ai", 'gagné', '.'], bleu,0.000
he's calm . => ['il', '<unk>', '.'], bleu,0.000
i'm home . => ['je', 'suis', 'certain', '.'], bleu,0.512
```

In this next experiment, we replaced the GRU model with LSTM, and then once again adjusted the number of hidden layers from 1-4. Overall, the training and validation results were comparable to the GRU version. The prediction results tended to be more skewed, however. This is where we can see the value of adjusting the number of layers. 1 layer yielded the best results, but increasingly adding additional layers yielded worse and worse predictions.

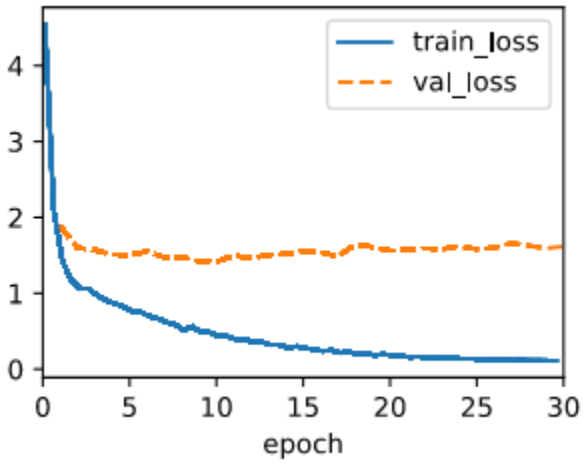
The baseline model:
The training results:



The prediction results:

```
go . => ['va', '!'], bleu,1.000
i lost . => ["j'ai", 'perdu', '.'], bleu,1.000
he's calm . => ['nous', '<unk>', '.'], bleu,0.000
i'm home . => ['je', 'suis', 'malade', '.'], bleu,0.512
```

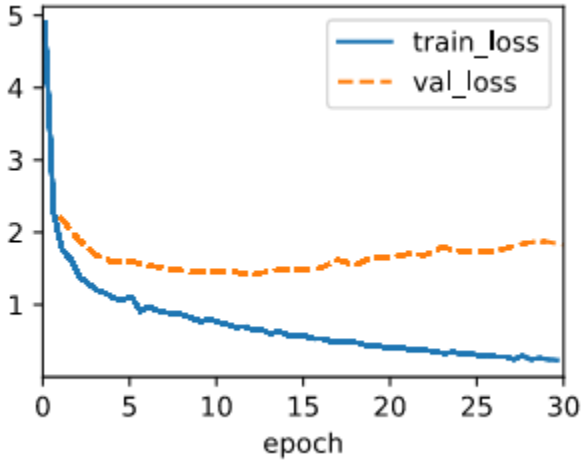
With 1 layer:
The training results:



The prediction results:

```
go . => ['va', '!'], bleu,1.000
i lost . => ["j'ai", 'perdu', '.'], bleu,1.000
he's calm . => ['il', 'est', 'mouillé', '.'], bleu,0.658
i'm home . => ['je', 'suis', 'chez', 'moi', '.'], bleu,1.000
```

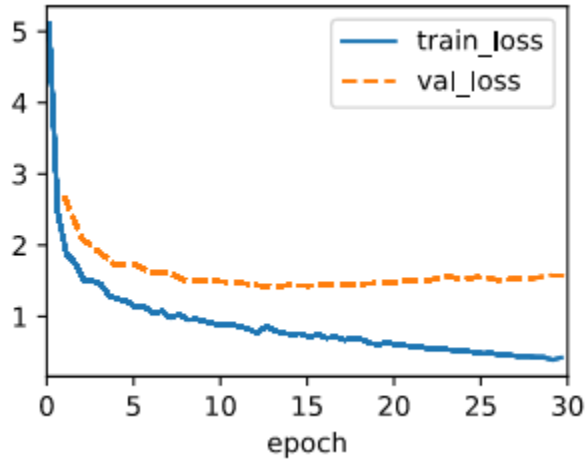
With 2 layers:
The training results:



The prediction results:

```
go . => ['<unk>', '.'], bleu,0.000
i lost . => ["j'ai", 'perdu', '.'], bleu,1.000
he's calm . => ['il', 'court', '.'], bleu,0.000
i'm home . => ['je', 'suis', 'chez', 'moi', '.'], bleu,1.000
```

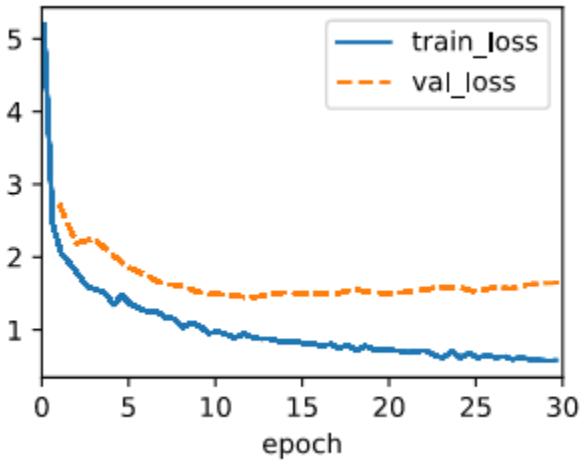
With 3 layers:
The training results:



The prediction results:

```
go . => ['<unk>', '.'], bleu,0.000
i lost . => ['j’ai', '<unk>', '.'], bleu,0.000
he's calm . => ['il', '<unk>', '<unk>', '.'], bleu,0.000
i'm home . => ['je', 'suis', '<unk>', '.'], bleu,0.512
```

With 4 layers:
The training results:



The prediction results:

```
go . => ['<unk>', '!'], bleu,0.000
i lost . => ['je', 'suis', '<unk>', '.'], bleu,0.000
he's calm . => ['il', '<unk>', 'emporté', '.'], bleu,0.000
i'm home . => ['je', 'suis', '<unk>', '.'], bleu,0.512
```